**NAME: AQSA TAUHEED (2303-KHI-DEG-011)**

**PAIRING WITH : MAVIA ALAM KHAN (2303.KHI.DEG.017)**

**&**

**MOHAMMAD HUSSAM(2033.KHI.DEG.020)**

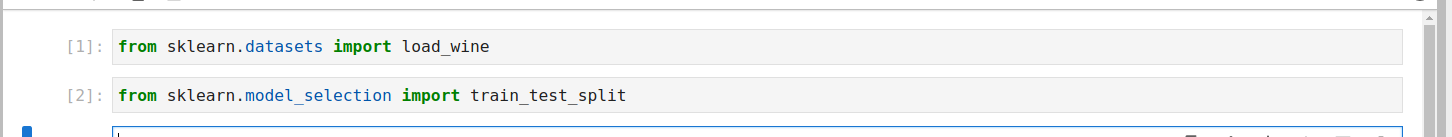
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ASSIGNMENT 3.2**

Implement a single classification model of your choice and try  
to achieve at least an 80% F1 score on the wine dataset  
provided by Sklearn.

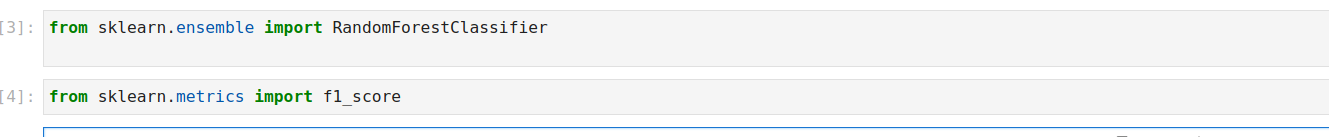
**SOLUTION:**

**STEP:1**  
Firstly we import libraries :



from sklearn.datasets import load\_wine : sklearn.datasets to load the wine dataset

from sklearn.model\_selection import train\_test\_split :to split the data into training and testing sets



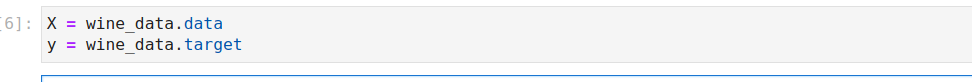
from sklearn.ensemble import RandomForestClassifier: to train a random forest classifier

from sklearn.metrics import f1\_score: to calculate the F1 score

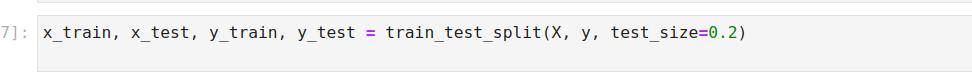
**STEP:2**

We loaded the wine dataset using load\_wine() function and assign it to the variable wine\_data

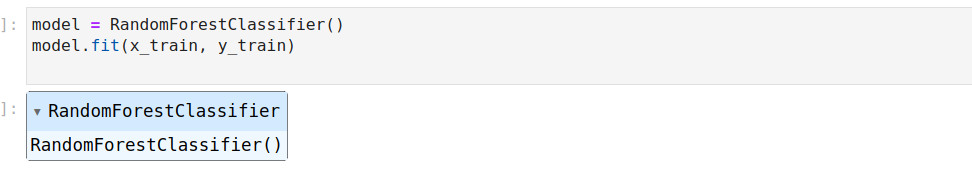
**STEP:3**

Splited the data into X and y variables, where X contains the feature data and y contains the target labels.

**STEP:4**

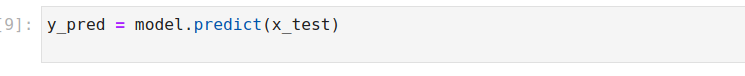
Splited the data into training and testing sets using the train\_test\_split function. This function randomly splits the data into training and testing sets based on the test\_size parameter (in this case, 20% of the data will be used for testing). The remaining 80% of the data will be used for training.

**STEP:5**

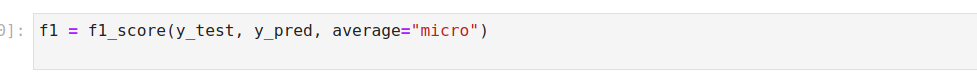
RandomForestClassifier() creates an instance of the Random Forest classifier model with default parameters.

fit(x\_train, y\_train) trains the model on the training data x\_train and their corresponding labels y\_train. During training, the model learns to map the input features to their corresponding class labels.

**STEP:6**

Make predictions on the testing data using the predict method and assign them to the variable y\_pred.

**STEP:7**



Calculate the F1 score using the f1\_score function. This function takes in the actual target labels (y\_test) and the predicted target labels (y\_pred) and returns the F1 score. In this case, we're using average="micro", which means we're calculating the F1 score for all classes combined.

**OUTPUT:**

Then we printed the F1 score as shown in image, which gives the F1 score 0.972222.. or 97.222 %

